Implémentation à seuil de boîtes S

Mathilde de La Morinerie

ÉCOLE POLYTECHNIQUE
UNIVERSITÉ PARIS-SACLAY

informatics mathematics

Stage de recherche - Rapport

Implémentation à seuil de boîtes S

DE LA MORINERIE Mathilde

Tutrice : CANTEAUT Anne
Référent : MORAIN François

Table des matières

Remerciements

Il est habituellement recommandé de terminer par l'essentiel, cependant je vais déroger à cette règle en commençant par les remerciements.

Tout d'abord je tiens à remercier chaleureusement ma tutrice Anne Canteaut pour son accueil, son accompagnement et sa disponibilité tout au long du stage. Tout en me laissant une grande autonomie, ses conseils et suggestions m'ont permis de surmonter bien des obstacles quant à la compréhension du sujet et aux difficultés de formalisation. J'ai énormement appris au cours de ce stage et j'aurais bien du mal à lui exprimer l'ampleur de ma reconnaissance.

Un grand merci également à Ferdinand, Matthieu et Valentin du bureau "Tapdance", qui m'ont acceptée dans leur processus de colonisation de la ZRR, tout en tolérant les déménagements internes successifs, et une certaine monopolisation du tableau. Qu'ils soient remerciés pour leur soutien technique et psychologique, leurs suggestions, et leur bonne humeur.

Je tiens aussi à remercier Sébastien, Xavier et Yann pour leurs cours accélérés sur les fonctions booléennes et les permutations qui m'ont fait gagner un temps précieux. Merci également de m'avoir fait parcourir le monde gustatif en m'acceptant dans leur palais des thés.

Par ailleurs, je remercie vivement les membres de l'équipe-projet SECRET Anne, Maria, Pascale, Jean-Pierre, Nicolas, Anthony, André, Gaëtan, Xavier, Kevin, Rodolfo, Kaushik, Thomas, Sebastien, Antoine, Vivien, Yann, André, les deux Matthieu, Valentin, Ferdinand, Christof et Sristi pour leur accueil, et leurs réponses patientes à mes nombreuses questions. J'ai grâce à eux découvert le monde de la recherche et de la cryptologie (et des masters associés), mais aussi appris les meilleures stratégies de babyfoot et de tarot.

Merci également à Gaëtan et à Ben pour m'avoir permis de reconsidérer mes aspirations de stage, et pour m'avoir à terme conduite jusqu'ici.

Enfin, ce stage n'aurait pas été tout à fait le même sans Etienne Chazal, Thierry Larsan et Bernard Phillipet qui ont égayé les pauses cafés en m'initiant aux techniques cruciverbistes. Merci à eux, et à Robert, pour leur apport de culture générale.

1 Introduction

La cryptographie, ou science du secret, a pour but principal de protéger l'information échangée entre un émetteur et un destinataire en présence d'adversaires. Elle se divise en deux domaines, la cryptographie à clé publique, et la cryptographie à clé secrète. Cette dernière, aussi appelée cryptographie symétrique, étudie des algorithmes de chiffrement de deux types : les algorithmes de chiffrement par flot, où chaque bit d'information est chiffré à la volée, et les algorithmes de chiffrement par blocs, où l'information est chiffrée par blocs de bits. C'est ce type de chiffrement qui nous intéressera par la suite.

Un chiffrement par blocs peut être décrit comme une succession de tours, chaque tour appliquant une étape de confusion, à l'aide d'une permutation non-linéaire appelée boîte S, puis une étape de diffusion linéaire. Chaque tour dépend également de la valeur d'une clé secrète.

Selon les pouvoirs accordés à l'adversaire, plusieurs types d'attaques sont possibles. Les attaques par canaux auxiliaires consistent à utiliser l'information donnée par l'implémentation physique d'un cryptosystème, comme le temps de calcul, le rayonnement électromagnétique, ou la consommation de courant électrique, pour retrouver tout ou partie du secret. Ces attaques sont non invasives et relativement faciles à mettre en place, et nécessitent donc de sécuriser les implémentations logicielles et matérielles par des contre-mesures efficaces. On se concentrera sur les attaques par étude de la consommation de courant et leurs contre-mesures dans ce qui suit.

Les algorithmes de chiffrement par blocs, et en particulier les permutations non-linéaires utilisées, sont sensibles aux attaques par canaux auxiliaires. Au niveau logiciel, les attaques par étude de la consommation de courant peuvent être contrées par du masquage sur les entrées. Cependant, en présence de glitches, cela ne suffit pas à protéger l'implémentation matérielle.

Pour sécuriser les implémentations matérielles des permutations non-linéaires, il est alors nécessaire d'allier le masquage des entrées avec du calcul multipartite. C'est le principe de l'implémentation à seuil. Malheureusement, toutes les boîtes S ne sont pas compatibles avec cette contre-mesure. Il est donc important de classifier celles qui peuvent être protégées, et de trouver la façon précise de les protéger.

C'est cette approche qui est développée par Bilgin et al. dans [BNN+15]. En utilisant une recherche exhaustive sur les solutions envisageables, ils fournissent une classification des boîtes S de petite taille compatibles avec les implémentations à seuil ayant un nombre de parts fixé.

Comme la recherche exhaustive est très coûteuse, elle ne peut pas être étendue aux cas plus grands. L'objectif de ce stage est de comprendre l'origine mathématiques des résultats obtenus par recherche exhaustive, afin de trouver des algorithmes plus efficaces pour classifier et protéger les boîtes S, et d'appliquer ces algorithmes aux boîtes S de plus grande taille.

Les résultats sont présentés dans ce qui suit. La section 2 introduit les notions théoriques nécessaires pour la suite, puis la section 3 résume de l'état de l'art. La section 4 présente les travaux qui ont été effectués au cours du stage. Enfin, la section 5 rassemble les problèmes restant pour la fin du stage.

2 Préliminaires – Implémentations matérielles de chiffrements par bloc

2.1 Boîtes S

Une *fonction booléenne* est une fonction de $\mathbb{F}_2^n$ dans $\mathbb{F}_2$. On peut la représenter par la table de ses valeurs, ou par un polynôme à n variables à coefficients dans $\mathbb{F}_2$ appelé *forme algébrique normale* de la fonction booléenne. Par la suite, l'opération $+$ dans la forme algébrique normale d'une fonction booléenne désignera l'addition dans $\mathbb{F}_2$ (XOR). On appelle *degré* d'une fonction booléenne le degré du plus grand monôme de sa forme algébrique normale. Le *poids de la fonction* est le nombre d'éléments de $\mathbb{F}_2^n$ ayant pour image 1.

Par exemple, la fonction de $\mathbb{F}_2^3$ dans $\mathbb{F}_2$ qui à (a, b, c) associe $ab + c$ est une fonction booléenne de degré 2, et de poids 4, car seuls les triplets $(1, 1, 0), (1, 0, 1), (0, 1, 1)$ et $(0, 0, 1)$ ont pour image 1. Sa table de vérité est représentée ci-dessous.

a	0	0	1	1	0	0	1	1
b	0	1	0	1	0	1	0	1
c	0	0	0	0	1	1	1	1
$ab + c$	0	0	0	1	1	1	1	0

TABLE 1 – Table de vérité de la fonction $ab + c$

Une fonction booléenne de $\mathbb{F}_2^n$ dans $\mathbb{F}_2$ est dite *équilibrée* si sa sortie est uniformément distribuée, c'est-à-dire si elle retourne autant de fois la valeur 0 que la valeur 1 lorsque les entrées parcourent l'ensemble de définition. Autrement dit, une fonction booléenne est équilibrée si et seulement si elle est de poids $2^{(n-1)}$.
La fonction donnée en exemple ci-dessus est équilibrée.

Une *substitution*, ou *boîte S* est une fonction de $\mathbb{F}_2^n$ dans $\mathbb{F}_2^n$, c'est-à-dire la donnée de n fonctions booléennes. Ces fonctions booléennes sont appelées *coordonnées booléennes*. Intuitivement, une substitution substitue chaque élément de $\mathbb{F}_2^n$ par un élément de $\mathbb{F}_2^n$, mais pas toujours bijectivement car deux éléments peuvent être envoyés sur la même valeur.
On note $(a, b, c, ...)$ le n-uplet d'entrée d'une boîte S, et

$$(f(a, b, c, ...), g(a, b, c, ...), h(a, b, c, ...), ...)$$

sa sortie, où f, g, h, ... sont n coordonnées booléennes de $\mathbb{F}_2^n$ dans $\mathbb{F}_2^n$.
Par exemple, la fonction de $\mathbb{F}_2^3$ dans $\mathbb{F}_2^3$ qui à (a, b, c) associe $(ac + b, ab, bc + a + b + c)$ est une substitution. Ses coordonnées booléennes sont $ac + b$, ab et $bc + a + b + c$.

En pratique, on considère le plus souvent des substitutions bijectives, appelées *permutations*. La propriété suivante permet de caractériser les permutations et sera très utile pour la suite.

Proposition 2.1.1 (Can16). *Une substitution est bijective, donc une permutation, si et seulement si toute combinaison linéaire de ses coordonnées booléennes est équilibrée.*

Démonstration. Voir [Can16], chapitre 2, Proposition 2.2. □

La substitution donnée en exemple ci-dessus n'est pas une permutation, car la coordonnée booléenne ab est de poids 2, donc pas équilibrée. En revanche, la substitution $(ac+b, ab+b+c, bc+a+b+c)$ est une permutation, car toute combinaison linéaire de ses coordonnées est équilibrée.

Dans la suite, on utilisera le terme "boîte S" pour désigner une permutation (plutôt qu'une substitution) car ce sont essentiellement les permutations qui seront étudiées.

2.2 Classes d'équivalence affine

Une *permutation affine* de $\mathbb{F}_2^n$ est une permutation de la forme $x \to Ax + b$, où A est une matrice de taille $n \times n$ inversible et b est un vecteur constant de $\mathbb{F}_2^n$.

Les permutations affines permettent de définir des classes d'équivalence appelées *classes affines*. Deux boîtes S notées S_1 et S_2 appartiennent à la même classe affine si et seulement s'il existe deux permutations affines P et Q telles que $S_1 = P \circ S_2 \circ Q$.

Concrètement, deux boîtes S sont dans la même classe d'équivalence si on peut passer de l'une à l'autre par changement de variable ou combinaisons linéaires sur les coordonnées booléennes.

Par exemple, $(a, b, ac+b)$, $(ac+a+b, b, ac+b)$ et $(b, c, ab+c)$ sont toutes les trois dans la même classe d'équivalence. En effet, on passe de la première fonction à la deuxième par combinaison linéaire de la première et de la troisième coordonnée ; et on passe de la première forme à la troisième par changement de variable cyclique (abc).

2.3 Attaques par canaux auxiliaires sur les chiffrements par blocs - Implémentations à seuil de boîtes S

Les attaques par canaux auxiliaires introduites par Kocher en 1996 dans [Koc96] consistent à utiliser la consommation de courant, le bruit ou les émissions électromagnétiques, voire à injecter des erreurs lors du calcul, pour en déduire des informations sur les données qui transitent. On se concentre ici sur les attaques par analyse de la consommation de courant (Simple and Differential Power Analysis). Comme ces attaques sont relativement faciles d'accès et peu coûteuses, elles représentent une vraie menace pour la sécurité. Il est donc essentiel de sécuriser les implémentations contre ces risques (voir [Sta10]).

Les chiffrements par blocs sont une des grandes familles de chiffrement symétrique. Un chiffrement par bloc consiste à enchaîner plusieurs *tours* d'une même permutation, qui dépend de la valeur d'une *clé* tenue secrète. C'est la connaissance de cette clé qui permet de chiffrer un message, et surtout de le déchiffrer. Il est donc indispensable de protéger la donnée de la clé contre les attaques par canaux auxiliaires, en particulier contre les attaques par étude de la consommation de courant.

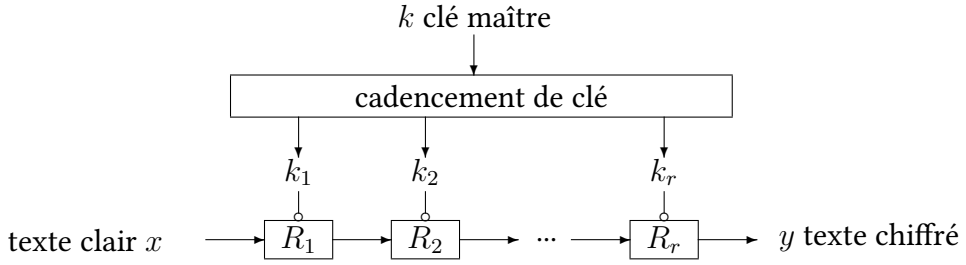

FIGURE 1 – Algorithme de chiffrement par bloc.

Une première façon de protéger les implémentations des chiffrements par blocs contre les attaques par étude de la consommation de courant est de découper les entrées en plusieurs valeurs aléatoires. Typiquement, une variable x peut être séparée en deux variables $x_1 = r$ et $x_2 = x + r$ avec r un vecteur aléatoire et l'addition dans $\mathbb{F}_2^n$, telles que $x_1 + x_2 = x$. Cette méthode, appelée *masquage booléen* ([CJRR99] [AG01] [GP99]), peut être généralisée pour découper une entrée en n variables, en utilisant $n-1$ vecteurs aléatoires. Ces découpages imposent de connaître toutes les valeurs x_i pour obtenir de l'information sur le x initial, ce que la simple étude de consommation de courant électrique ne permet pas. Le masquage booléen est donc une contre-mesure efficace tant que chaque tour est censé agir comme une boîte noire, c'est-à-dire en supposant qu'il n'est pas possible d'avoir accès à ce qu'il se passe au cours du tour, mais seulement entre les tours.

Malheureusement, les tours successifs n'agissent pas toujours exactement en boîte noire. Les contraintes des implémentations matérielles imposent des délais de propagation de l'information, appelés *glitches*, et tous les résultats n'arrivent pas en même temps en sortie du tour. En fait, on peut même provoquer ces délais en faisant varier brutalement la tension imposée au circuit. Ces retards permettent d'avoir accès à des états intermédiaires qui peuvent fournir beaucoup d'information sur les données qui transitent.

Une telle attaque a été mise en évidence pour la première fois en 2005 par Mangard, Pramstaller et Oswald. Ils ont montré dans [MPO05] que le masquage n'était pas suffisant pour protéger les implémentations matérielles de l'AES en présence de glitches.

Une solution pour pallier ce problème est d'utiliser des *implémentations à seuil*, c'est-à-dire à la fois du masquage sur les entrées, mais aussi du calcul multipartite sur les coordonnées booléennes de la permutation évaluée par le chiffrement. Cette technique a été proposée pour la première fois en 2011 par Nikova, Rechberger et Rijmen dans [NRR06].

Concrètement, les implémentations à seuil consistent à découper les entrées, mais aussi chaque coordonnée booléenne en k parts, tout en conservant certaines propriétés, de façon à obtenir une permutation de taille $nk \times nk$ résistante aux attaques en présence de glitches.

Plus formellement, l'implémentation à seuil d'une boîte S $n \times n$ consiste d'abord à découper chacune des n entrées $(a, b, c, ...)$ en k parts $(a_1, a_2, ..., a_k, b_1, b_2, ..., b_k, c_1, c_2, ..., c_k, ...)$ telles que

$$a = \sum_{i=1}^{k} a_i, b = \sum_{i=1}^{k} b_i, c = \sum_{i=1}^{k} c_i, ...$$

où la somme représente une somme dans $\mathbb{F}_2$. Cela peut être facilement fait en utilisant $k-1$ valeurs aléatoires pour a_1, ..., a_{k-1}, puis en posant $a_k = a + \sum_{i=0}^{k-1} a_i$.

Les coordonnées booléennes $(f, g, h, ..)$ de la permutation vont elles aussi être découpées en k parts de façon à avoir :

$$f = \sum_{i=1}^{k} f_i, g = \sum_{i=1}^{k} g_i, h = \sum_{i=1}^{k} h_i, \ldots .$$

Le découpage des coordonnées booléennes doit satisfaire trois propriétés dans le cadre des implémentations à seuil :

Correction : Pour toute entrée $(a, b, c, ...)$ et pour toute coordonnée booléenne f de la boîte S, on a $f(a, b, c, ...) = \sum_{i=1}^{k} f_i(a_1, a_2, ...a_k, b_1, b_2, ..., b_k, c_1, c_2, ..., c_k, ...)$.

Indépendance : Pour chaque coordonnée booléenne f, g, h, ..., les parts f_i, g_i, h_i,... doivent être indépendantes des entrées d'indice i, i.e. de a_i, b_i, c_i, ... Cela revient à dire qu'une fonctions d'indice i ne prend pas $a_i, b_i, c_i, ...$ en entrée.

Équilibre : Toute combinaison linéaire des parts doit être équilibrée.

Les trois propiétés sont nécessaires pour obtenir une permutation de taille $nk \times nk$ résistante aux attaques en présence de glitches. La propriété de correction permet d'assurer que le résultat final avec l'implémentation à seuil sera identique au résultat de la boîte S initiale.
En supposant qu'un attaquant ne peut mesurer la consommation de courant qu'en un seul point au cours d'un tour, la propriété d'indépendance assure que cet attaquant ne sera pas en mesure d'obtenir d'information sur les entrées, puisque les k parts d'une même variable sont nécessaires pour obtenir de l'information sur cette variable. Chaque partie du calcul est donc indépendante des entrées de la boîte S, ce qui assure qu'il n'y a pas de fuite d'information sur les entrées, même en présence de glitches.
La propriété d'équilibre assure que la boîte S $nk \times nk$ obtenue après découpage sera bien inversible, mais aussi itérable plusieurs fois, et donc utilisable dans un chiffrement par blocs. Si la propriété d'équilibre n'est pas vérifiée, il faut effectuer un nouveau masquage, avec introduction d'aléa, avant d'effectuer le tour suivant. C'est aussi la propriété la plus délicate à obtenir.

L'implémentation à seuil d'une boîte S $n \times n$ sera donc la donnée d'une nouvelle permutation $kn \times kn$, puisque chaque entrée et chaque fonction aura été découpée en k parts et en k fonctions respectivement. En respectant les trois propriétés, les implémentations à seuil permettent de sécuriser des systèmes de chiffrements par blocs contre les attaques par analyse de la consommation de courant, même en présence de glitches. Cependant, la décomposition des coordonnées booléennes en parts respectant les contraintes ci-dessus est loin d'être évidente.

3 État de l'art

Deux articles successifs, [BNN+12] et [BNN+15], étudient les implémentations à seuil des boîtes S de petite taille ($n = 3, 4$ et 5) avec peu de parts ($k = 3$ ou 4). Leur étude repose sur une classification empirique des boîtes S en classes d'équivalence affine, en utilisant notamment les travaux de De Cannière qui fournissent un représentant pour chacune des classes d'équivalence affine des boîtes S 3×3 et 4×4 ([De 07]). Ils montrent alors par recherche

exhaustive que certaines classes d'équivalence sont compatibles avec une implémentation à seuil alors que d'autres ne le sont pas.

Les détails de leurs travaux sont résumés dans ce qui suit.

3.1 Nombre de parts nécessaires pour les implémentations à seuil

Nikova, Rijmen et Schläffer donnent dans [NRS11] une borne inférieure sur le nombre de parts nécessaires au découpage d'une fonction.

Théorème 3.1. *Au moins $d+1$ parts sont nécessaires pour réaliser un découpage d'une fonction booléenne de degré d vérifiant les propriétés de correction et d'indépendance.*

Démonstration. Soit f une fonction booléenne de degré d.
La forme algébrique normale de cette fonction possède un terme $abc...x$ de degré d. Après masquage des entrées, on obtient donc un monôme de la forme $a_1b_2c_3...x_d$. D'après la propriété de correction, ce terme doit être présent dans une des parts. Or d'après la propriété d'indépendance, chaque part f_i doit être indépendante des entrées d'indice i, c'est-à-dire de a_i, b_i, c_i, etc. Le terme $a_1b_2c_3...x_d$ ne peut donc être présent que dans une part f_{d+k}, avec $k \geqslant 1$.
Donc au moins $d+1$ parts sont nécessaires au découpage d'une fonction booléenne de degré d.

□

3.2 Classes d'équivalence affine de boîtes S 3×3 et restriction à un seul représentant

Les travaux de De Cannière [De 07] montrent qu'il existe quatre classes d'équivalence pour les boîtes S 3×3, notées A_1^3, Q_1^3, Q_2^3, Q_3^3 et donnent un représentant de chacune de ces classes. Ces représentants sont ici définis par la liste des huit valeurs prises par la fonction, en identifiant $\mathbb{F}_2^3$ et les entiers $\{0, ..., 7\}$:

- la permutation dont la table de valeur est 01234567 représente la classe A_1^3.
  C'est l'identité ;
- la permutation 01234576 (aussi notée (67) en représentation cyclique) représente la classe Q_1^3 ;
- la permutation 01234675 (aussi notée (567)) représente la classe Q_2^3 ;
- la permutation 01243675 (aussi notée $(34)(567)$) représente la classe Q_3^3.

D'après [BNN+15], la recherche d'un découpage correct, indépendant et équilibré des coordonnées booléennes pour tous les éléments d'une classe d'équivalence se restreint à la recherche d'un découpage correct, indépendant et équilibré d'un seul représentant quelconque d'une classe.

En effet, par définition des classes affines, on passe d'un représentant à un autre par changements de variables et par combinaisons linéaires entre les coordonnées booléennes.
Dans le cas d'un changement de variable, on peut appliquer le même changement de variable à chaque a_i, b_i et c_i. Par exemple, lors d'un changement $a \to a + b$, le découpage sera obtenu par les changements de variables $a_i \to a_i + b_i$.

Dans le cas de combinaisons linéaires des sorties, on peut appliquer les mêmes combinaisons aux parts selon chaque indice. Ainsi, une combinaison $f + g$ donnera un nouveau découpage $f_i + g_i$.
Comme l'équilibre est préservé par permutation, on obtient bien un découpage pour n'importe quel élément de la classe d'équivalence à partir du découpage d'un représentant.

3.3 Algorithme de *découpage direct* et termes correctifs

Pour un partage en $k = 3$ parts, une méthode algorithmique décrite dans [BNN+12] permet d'obtenir un découpage des coordonnées booléennes, en garantissant d'avoir la correction et l'indépendance. En revanche cet algorithme ne garantit pas l'équilibre.

L'algorithme commence par remplacer les entrées (a, b, c) par la somme de leurs parts, i.e. par $(a_1 + a_2 + a_3, b_1 + b_2 + b_3, c_1 + c_2 + c_3)$, dans les coordonnées booléennes. Par exemple, une coordonnée $f = ab + c$ deviendra

$$f = (a_1 + a_2 + a_3)(b_1 + b_2 + b_3) + c_1 + c_2 + c_3$$
$$f = a_1b_1 + a_1b_2 + a_2b_1 + a_2b_2 + a_2b_3 + a_3b_2 + a_3b_3 + a_3b_1 + a_1b_3 + c_1 + c_2 + c_3 \ .$$

On cherche maintenant à répartir chaque terme de l'expression ci-dessus dans trois fonctions f_1,f_2 et f_3 telles que $f = f_1 + f_2 + f_3$ (propriété de correction), et telles que la fonction f_i ne dépende pas de a_i, b_i et c_i (propriété d'indépendance).
La propriété d'indépendance impose de placer les termes de la forme a_ib_{i+1} dans la fonction f_{i+2}. Les termes de la forme a_ib_i et c_i peuvent être placés dans f_{i+1} ou f_{i+2}. Par convention, on les placera dans f_{i+2}.
On obtient donc :

- $f_1 = a_2b_2 + a_2b_3 + a_3b_2 + c_2$;
- $f_2 = a_3b_3 + a_3b_1 + a_1b_3 + c_3$;
- $f_3 = a_1b_1 + a_1b_2 + a_2b_1 + c_1$.

On peut noter que dans le cas des fonctions linéaires, le découpage direct avec plus de deux parts fournit une implémentation à seuil équilibrée, puisqu'une fonction linéaire est toujours équilibrée. Cependant dans le cas général, ce découpage ne garantit pas d'avoir toute combinaison linéaire non triviale équilibrée.

Dans le cas où l'algorithme de découpage direct ne donne pas une permutation $3n \times 3n$, c'est-à-dire s'il existe des combinaisons linéaires non équilibrées, il est nécessaire d'ajouter des *termes correctifs* aux parts pour rétablir l'équilibre et obtenir un découpage équilibré. Afin de préserver la propriété de correction, ces termes correctifs doivent être ajoutés en nombre pair, et répartis sur les parts d'une même coordonnée booléenne. Ainsi un terme correctif dans le cas $k = 3$ doit être ajouté à deux parts, donc il ne peut s'agir que de termes de la forme a_i, b_i, c_i, ou a_ib_i, a_ic_i, b_ic_i.

3.4 Recherche exhaustive d'une implémentation à seuil

Pour chercher exhaustivement un découpage, il faut essayer tous les termes correctifs possibles sur chaque part, puis vérifier que toutes les combinaisons linéaires de parts sont bien équilibrées. C'est la méthode présentée par [BNN+15].

Soit f une coordonnée booléenne d'une permutation de taille n. On souhaite découper f en trois parts f_1, f_2, et f_3. Pour l'ensemble des trois parts, il y a $3n$ termes correctifs linéaires (les $a_i, b_i, \ldots$ pour $1 \leqslant i \leqslant 3$) et $3\binom{n}{2}$ termes correctifs quadratiques (les $a_i b_i, a_i c_i, \ldots$ pour $1 \leqslant i \leqslant 3$). Il y a donc $2^{3\left(n+\binom{n}{2}\right)}$ possibilités de découpage pour une coordonnée booléenne.

En considérant les n coordonnées booléennes, on obtient $2^{n\left(3n+3\binom{n}{2}\right)} = 2^{3n^2+3n\binom{n}{2}}$ possibilités pour le découpage des n coordonnées de la permutation.

Pour les boîtes S 3×3, cela fait 2^{54} possibilités à tester, et 2^{192} pour les boîtes S 4×4, ce qui est hors de portée.

De plus, pour chaque possibilité de découpage des n coordonnées, il faut vérifier si les $(2^{3n} - 1)$ combinaisons linéaires des parts sont équilibrées.

La recherche exhaustive devient donc rapidement impraticable, même pour des permutations ayant peu de variables.

3.5 Compatibilité des classes affines avec l'implémentation à seuil

Les auteurs de [BNN+15] ont sélectionné un représentant de chaque classe d'équivalence de boîtes S 3×3 de degré 2, et ont observé les résultats suivants :

- la classe Q_1^3 admet un découpage en appliquant l'algorithme de découpage direct ;
- la classe Q_2^3 admet un découpage en appliquant l'algorithme de découpage direct, puis en ajoutant des termes correctifs ;
- la classe Q_3^3 n'admet aucun découpage équilibré.

Ces résultats ont été obtenus en essayant toutes les corrections possibles sur les représentants de chaque classe. Cela fournit une correction pour le représentant de Q_2^3, en revanche aucune correction ne rend toutes les combinaisons linéaires de parts de Q_3^3 équilibrées.

4 Implémentation à seuil de boîtes S 3×3 avec 3 parts

L'un des objectifs du stage a été de comprendre pourquoi certaines classes d'équivalence des boîtes S 3×3 admettent un découpage équilibré en trois parts et pas d'autres. Un autre objectif a été de trouver un algorithme permettant de trouver une implémentation à seuil, ou de prouver qu'il n'en existe pas, plus efficacement que la recherche exhaustive utilisée dans [BNN+12] et [BNN+15]. C'est ce qui est décrit dans cette section.

4.1 Classes d'équivalence et représentants simplifiés

Rappelons que les permutations de $\mathbb{F}_2^3$ se répartissent en quatre classes d'équivalence :

- la permutation 01234567 représente la classe A_1^3 ;
- la permutation 01234576 représente la classe Q_1^3 ;
- la permutation 01234675 représente la classe Q_2^3 ;
- la permutation 01243675 représente la classe Q_3^3.

La formule de Möbius (voir [Can16], section 1.1.2) permet d'obtenir les formes algébriques normales de ces permutations. On trouve alors que :

- (a, b, c) est un représentant de A_1^3 ;
- $(a, b, ab + c)$ est un représentant de Q_1^3 ;
- $(a, ac + b, ab + ac + c)$ est un représentant de Q_2^3 ;
- $(ab + ac + bc, ab + bc + a + b, bc + a + c)$ est un représentant de Q_3^3.

La classe correspondant à l'identité n'ayant pas d'intérêt, nous nous contenterons d'étudier Q_1^3, Q_2^3 et Q_3^3 dans la suite. On notera (f, g, h) ces représentants, où f, g et h sont les trois coordonnées booléennes.

En appliquant des permutations affines, qui laissent la classe affine invariante, on peut simplifier ces représentants. Ainsi, on obtient pour Q_2^3 un nouveau représentant $(a, ac+b, ab+b+c)$ qui correspond à $(f, g, g+h)$. De même, avec les combinaisons linéaires $(f+g, g+h, h)$ puis un changement de variable $c = c + 1$, on obtient pour Q_3^3 le représentant $(ac + b, ab + b + c, bc + a + b + c)$.

On travaillera donc avec les représentants simplifiés suivants :

- $(a, b, ab + c)$ représente la classe Q_1^3 ;
- $(a, ac + b, ab + b + c)$ représente la classe Q_2^3 ;
- $(ac + b, ab + b + c, bc + a + b + c)$ représente la classe Q_3^3.

On remarque que les représentants simplifiés de Q_1^3, Q_2^3 et Q_3^3 ne possèdent respectivement qu'un, deux ou trois termes quadratiques différents, ce qui donne une intuition de l'origine des différents comportements de ces classes d'équivalence. C'est précisement cette différence dans le nombre de termes quadratiques qui va expliquer pourquoi certaines classes sont découpables et d'autres pas.

4.2 La classe affine Q_1^3

D'après [BNN+15], les éléments de la classe affine Q_1^3 admettent un découpage qui permet de les utiliser dans des implémentations à seuil. Ce résultat se fonde sur une recherche heuristique par ordinateur, et le but de cette section est de l'expliquer mathématiquement.

Comme mentionné précédemment, trouver un découpage pour un représentant d'une classe d'équivalence permet d'obtenir un découpage pour l'ensemble des éléments de la classe. On va donc se concentrer sur le représentant simplifié de Q_1^3, à savoir $(a, b, ab + c)$, noté aussi (f, g, h) dans la suite pour alléger les notations. On cherche donc un découpage des trois coordonnées booléennes (f, g, h) en neuf fonctions booléennes $((f_i)_{1\leqslant i\leqslant 3}, (g_i)_{1\leqslant i\leqslant 3}, (h_i)_{1\leqslant i\leqslant 3})$ vérifiant les trois propriétés nécessaires aux implémentations à seuil, à savoir la correction, l'indépendance et l'équilibre.

L'algorithme de *découpage direct* décrit précédemment permet d'obtenir un découpage vérifiant les propriétés de correction et d'indépendance. Les neuf fonctions obtenues par découpage direct sont :

$$
\begin{aligned}
f_1 &= a_2 \\
f_2 &= a_3 \\
f_3 &= a_1 \\
g_1 &= b_2 \\
g_2 &= b_3 \\
g_3 &= b_1 \\
h_1 &= a_2b_2 + a_2b_3 + a_3b_2 + c_2 \\
h_2 &= a_3b_3 + a_3b_1 + a_1b_3 + c_3 \\
h_3 &= a_1b_1 + a_1b_2 + a_2b_1 + c_1 \ .
\end{aligned}
$$

On a $f_i = a_{i+1}$, $g_i = b_{i+1}$ et $h_i = a_{i+1}b_{i+1} + a_{i+1}b_{i+2} + a_{i+2}b_{i+1} + c_{i+1}$.

En réalité, ce découpage est aussi équilibré. Cela repose sur le fait que toute combinaison linéaire des neuf fonctions ci-dessus possèdera seulement des termes linéaires (comme $f_1 + f_2 + g_3$), ou un seul *type* de terme quadratique, c'est-à-dire des termes quadratiques reposant seulement sur les lettres a et b. Une somme de termes linéaires étant toujours équilibrée, on se concentre sur les combinaisons linéaires mettant en jeu des termes quadratiques.

Dans ce cas, l'équilibre repose sur la propriété suivante, présentée au chapitre 13 de [MS77] :

Proposition 4.2.1. *Si $f(v_1, v_2, ..., v_m)$ est une fonction booléenne, alors $f(v_1, v_2, ..., v_m) + v_{m+1}$ est une fonction équilibrée.*

Démonstration. Les deux cas $v_{m+1} = 0$ et $v_{m+1} = 1$ divisent la répartition des zéros et des uns de la fonction f en deux distributions identiques car v_{m+1} est indépendant des termes apparaissant dans f . En particulier ces deux distributions ont le même poids. Ainsi, l'ajout de v_{m+1} va inverser les zéros et les uns d'une seule de ces distributions, et donc rétablir l'équilibre.

Plus formellement, si on note $g(v_1, ..., v_{m+1}) = f(v_1, ..., v_m) + v_{m+1}$, alors

$$\begin{aligned} wt(g) &= \#\{(v_1, ..., v_m) : g(v_1, ..., v_m, 0) = 1\} + \#\{(v_1, ..., v_m) : g(v_1, ..., v_m, 1) = 1\} \\ &= wt(f) + wt(f+1) \\ &= wt(f) + 2^m - wt(f) \\ &= 2^m \ . \end{aligned}$$

La fonction g est donc bien équilibrée. □

Un exemple est probablement plus parlant que la démonstration : prenons la fonction $ab + c$, dont la table de vérité est représentée à la table 2. Le poids de la fonction ab est 2, cette fonction n'est donc pas équilibrée. Mais la distribution des zéros et des uns est identique pour les deux cas $c = 0$ et $c = 1$. Ainsi l'ajout de c va inverser la moitié des zéros et la moitié des uns exactement, et rétablir l'équilibre de la fonction. On a bien $ab + c$ de poids 4, donc l'ajout du terme indépendant c a rétabli l'équilibre.

a	0	0	1	1	0	0	1	1
b	0	1	0	1	0	1	0	1
c	0	0	0	0	1	1	1	1
ab	0	0	0	1	0	0	0	1
$ab + c$	0	0	0	1	**1**	**1**	**1**	**0**

TABLE 2 – Table de vérité de la fonction $ab + c$

Avec cette propriété, et en remarquant qu'un terme indépendant dans une coordonnée booléenne est préservé par découpage direct, on obtient que chaque part de la coordonnée h est équilibrée. En effet, chaque part h_i contient un élément c_{i+1} indépendant du reste de l'expression. Ainsi, les combinaisons linéaires contenant un terme quadratique, c'est-à-dire les combinaisons impliquant une ou plusieurs parts h_i, mettent en jeu un ou plusieurs termes indépendants c_i. Toutes les combinaisons linéaires sont donc équilibrées, et on a bien un découpage satisfaisant les trois propriétés demandées.

La classe affine Q_1^3 est donc compatible avec les implémentations à seuil, et le découpage direct fournit un algorithme pour trouver un découpage correct, indépendant et équilibré.

4.3 La classe affine Q_2^3

Les auteurs de [BNN+15] ont montré que le découpage direct de la classe affine Q_2^3 n'est pas équilibré, mais que l'ajout de termes correctifs permettait d'obtenir un découpage équilibré. Ce résultat provient à nouveau d'une recherche exhaustive par ordinateur et cette section a pour but de l'expliquer mathématiquement. Nous proposerons également un algorithme pour vérifier plus rapidement que par l'algorithme naïf si l'ajout d'un nombre donné de termes correctifs linéaires suffit à rendre le découpage direct équilibré.

Rappelons tout d'abord que le représentant utilisé pour la classe affine Q_2^3 est

$$(f, g, h) = (a, ac + b, ab + b + c) \ .$$

Contrairement à Q_1^3, ce représentant contient deux types de termes quadratiques : ab et ac. C'est cette différence qui est à l'origine de la perte d'équilibre lors du découpage direct. En effet, la présence de deux types de termes quadratiques donne naissance à des combinaisons linéaires des parts ne faisant pas intervenir de termes indépendants, contrairement à Q_1^3. Dans certains de ces cas, cela donnera une combinaison linéaire non-équilibrée.

Voici le découpage obtenu par découpage direct du représentant de Q_2^3 :

$$\begin{aligned}
f_1 &= a_2 \\
f_2 &= a_3 \\
f_3 &= a_1 \\
g_1 &= a_2c_2 + a_2c_3 + a_3c_2 + b_2 \\
g_2 &= a_3c_3 + a_3c_1 + a_1c_3 + b_3 \\
g_3 &= a_1c_1 + a_1c_2 + a_2c_1 + b_1 \\
h_1 &= a_2b_2 + a_2b_3 + a_3b_2 + b_2 + c_2 \\
h_2 &= a_3b_3 + a_3b_1 + a_1b_3 + b_3 + c_3 \\
h_3 &= a_1b_1 + a_1b_2 + a_2b_1 + b_1 + c_1 \ .
\end{aligned}$$

On remarque que la combinaison linéaire

$$\begin{aligned}
h_1 + g_2 + g_3 + h_3 = {} & a_2b_2 + a_2b_3 + a_3b_2 + a_1b_1 + a_1b_2 + a_2b_1 \\
& + a_3c_3 + a_3c_1 + a_1c_3 + a_1c_1 + a_1c_2 + a_2c_1 \\
& + b_2 + b_3 + c_2 + c_1
\end{aligned}$$

ne possède pas de terme indépendant des variables intervenant dans les termes quadratiques. Par ailleurs, cette fonction n'est effectivement pas équilibrée. Ceci suffit à prouver que le découpage direct dans Q_2^3 ne permet pas d'obtenir un découpage équilibré. (En revanche, l'équation similaire $g_1 + h_2 + g_3 + h_3$ est équilibrée, même si elle ne possède pas de terme linéaire indépendant. Il n'y a donc qu'une implication entre la présence de termes indépendants et l'équilibre).

Puisque le découpage direct n'est pas un découpage équilibré, on cherche des *termes correctifs* à ajouter aux différentes parts afin d'obtenir un découpage compatible avec les implémentations à seuil. De plus, il apparaît que la présence ou l'absence de termes indépendants ne permet pas de conclure sur l'équilibre d'un découpage dans Q_2^3. Un nouveau critère est donc nécessaire. Pour cela, nous aurons besoin des notions de *matrice de représentation* et de *dérivée*, qui sont introduites dans ce qui suit.

4.3.1 Représentation matricielle

Pour représenter les différentes fonctions de degré 2 obtenues en combinant les différentes parts, on peut utiliser une *matrice de représentation*, plus visuelle que la forme algébrique normale.

Définition 1. *Soit f une fonction booléenne à n variables de degré 2. Sa matrice de représentation, notée M_f, est la matrice binaire $n \times n$ dont le coefficient d'indice ij vaut 1 si et seulement si le terme x_ix_j intervient dans la forme algébrique normale de $f(x_1, x_2, ..., x_n)$.*

D'après cette définition, la matrice est symétrique.

Par exemple, la matrice correspondant aux termes quadratiques de la fonction $h_1 + g_2 + g_3 + h_3$ est donnée par :

$$M_{h_1+g_2+g_3+h_3} = \begin{pmatrix} 0 & 0 & 0 & 1 & 1 & 0 & 1 & 1 & 1 \\ 0 & 0 & 0 & 1 & 1 & 1 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 & 1 & 0 & 1 \\ 1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 1 & 1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 1 & 1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 1 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \end{pmatrix}.$$

Pour rendre cette matrice plus lisible, on ne conserve que les 1, les 0 étant remplacés par des blancs. Par ailleurs, on utilisera un tableau pour mettre en évidence les différents blocs.

	a_1	a_2	a_3	b_1	b_2	b_3	c_1	c_2	c_3
a_1				1	1		1	1	1
a_2				1	1	1	1		
a_3					1		1		1
b_1	1	1							
b_2	1	1	1						
b_3		1							
c_1	1	1	1						
c_2	1								
c_3	1		1						

TABLE 3 – Matrice de représentation de la fonction $h_1 + g_2 + g_3 + h_3$

De façon générale, on obtient dans Q_2^3 avec le représentant choisi une matrice symétrique, définie par bloc, du type :

0	bloc ab	bloc ac
bloc ab	0	0
bloc ac	0	0

TABLE 4 – Matrice type de Q_2^3

Les blocs de la même couleur sont identiques. Les blocs oranges correspondent aux termes quadratiques de type $a_i b_j$ issus des parts de la coordonnée h, et les blocs en jaunes correspondent aux termes quadratiques de type $a_i c_j$ issus des parts de la coordonnée g. Tous les autres termes sont nuls pour le représentant choisi. De plus, dans le découpage direct, chaque bloc coloré est obtenu par addition des trois blocs de base issus des parts respectives de chaque coordonnée, conformément à la définition du découpage direct, à savoir :

0	0	0
0	1	1
0	1	0

0	0	1
0	0	0
1	0	1

1	1	0
1	0	0
0	0	0

TABLE 5 – Blocs correspondant aux parts g_1, h_1 (à gauche), g_2, h_2 (au milieu) et g_3, h_3 (à droite).

En éliminant le cas d'un bloc nul, il n'y a donc que sept possibilités pour chaque bloc, qui correspondent aux différentes combinaisons linéaires. De plus, chacune de ces possibilités donne un bloc symétrique.

Remarquons que la propriété de symétrie est conservée pour n'importe quel découpage puisque l'ajout de termes correctifs quadratiques correspond à la modification d'éléments sur la diagonale des blocs.

La matrice de représentation d'une fonction quadratique a donc une structure symétrique et creuse, définie par blocs symétriques, et permet de représenter plus simplement l'ensemble des combinaisons linéaires de parts possibles.

4.3.2 Dérivée

L'absence de termes indépendants dans certaines combinaisons linéaires de parts demande de trouver un autre critère pour déterminer si une fonction est équilibrée. C'est la notion de *dérivée de fonction booléennes* qui va permettre de résoudre ce problème.

Définition 2. *Soit F est une fonction booléenne de $\mathbb{F}_2^n$. On appelle* dérivée de F par rapport au point $\alpha \in \mathbb{F}_2^n$ *la fonction*

$$\begin{aligned} D_\alpha F(x_1, ..., x_n) &= D_{(\alpha_1,...,\alpha_n)} F(x_1, ..., x_n) \\ &= F(x_1 + \alpha_1, ..., x_n + \alpha_n) + F(x_1, ..., x_n) \ . \end{aligned}$$

Par exemple, la dérivée de $h_1 = a_2b_2 + a_2b_3 + a_3b_2 + b_2 + c_2$ par rapport au point $(u_1, u_2, u_3, v_1, v_2, v_3, w_1, w_2, w_3)$ est :

$$\begin{aligned} D_{(u_1,...,w_3)} h_1(a_1, ..., c_3) &= h_1(a_1 + u_1, ..., c_3 + w_3) + h_1(a_1, ..., c_3) \\ &= (a_2 + u_2)(b_3 + v_3) + (a_3 + u_3)(b_1 + v_1) + (a_2 + u_2)(b_3 + v_3) \\ &\quad + b_2 + v_2 + c_2 + w_2 \\ &\quad + a_2b_2 + a_2b_3 + a_3b_2 + b_2 + c_2 \\ &= a_2(v_2 + v_3) + a_3v_2 + b_2(u_2 + u_3) + b_3u_2 \\ &\quad + u_2v_2 + u_2v_3 + u_3v_2 + v_2 + w_2 \ . \end{aligned}$$

Dans le cas où F est de degré 2, et c'est le cas pour toute combinaison linéaire des coordonnées de Q_2^3, les dérivées ont des propriétés particulières qui vont permettre de déterminer si F est équilibrée. Ces propriétés sont énoncées et démontrées dans ce qui suit. Les deux premières propositions (4.3.1 et 4.3.2) permettent d'obtenir la troisième, qui se simplifie dans le cas particulier de Q_2^3 pour aboutir à la propriété 4.3.4.

Dans toute la suite, on suppose que $F(0) = 0$. En effet, l'ajout d'une constante à F ne modifie pas ses propriétés de découpage, ni son caractère équilibré. On étudiera donc toujours un représentant tel que $F(0) = 0$

Proposition 4.3.1. *Une fonction booléenne F de degré 2 est équilibrée si et seulement s'il existe un point α tel que la dérivée de la fonction par rapport à ce point soit égale à 1. Formellement :*

$$F \text{ est équilibrée} \iff \exists \alpha \in \mathbb{F}_2^n, D_\alpha F = 1 \ .$$

Démonstration.

La démonstration est détaillée en annexe. Elle introduit la notion de transformée de Walsh et un lemme supplémentaire. En voici un bref résumé.

Le point clé est le fait que pour les fonctions booléennes de degré 2, la dérivée est de degré 1, et la transformée de Walsh donne alors $\varepsilon(F)^2 = \sum_{\alpha, D_\alpha F = 0} \varepsilon(D_\alpha F) + \sum_{\alpha, D_\alpha F = 1} \varepsilon(D_\alpha F)$.

Cette équation permet alors de relier l'équilibre d'une fonction booléenne à l'existence de points par rapport auxquels la dérivée est non nulle, et d'obtenir le résultat.

□

Cette propriété peut être améliorée en utilisant la proposition suivante, qui restreint l'ensemble des α possibles.

Proposition 4.3.2. *La dérivée d'une fonction booléenne F de degré 2 par rapport à un point α est constante si et seulement si α appartient au noyau de la matrice de représentation de F. Dans ce cas, en prenant $F(0) = 0$, la dérivée en α est égale à $F(\alpha)$. Formellement :*

$$\begin{aligned} D_\alpha F \text{ est constante} &\iff \alpha \in \ker(M_f) \\ &\iff D_\alpha F = F(\alpha) \ . \end{aligned}$$

Démonstration. Pour obtenir une dérivée constante, les termes x_i ne doivent pas apparaitre dans l'expression de la dérivée, ce qui impose que leurs facteurs soient nuls. Cela donne certaines contraintes sur les valeurs de $(\alpha_1, ..., \alpha_n)$. En les écrivant dans une matrice, on obtient exactement la matrice de représentation de la fonction F, par définition de la dérivée. Les points α tels que la dérivée de F soit constante sont donc dans le noyau de la matrice de représentation de F, notée M_f. Notons que la matrice M_f étant symétrique, le noyau est indifféremment le noyau à gauche ou à droite.

Formellement, avec ε_{ij} et $\zeta_i \in \{0, 1\}$ et en notant

$$F(x_1, ..., x_n) = \sum_{0<i<j\leq n} \varepsilon_{ij} x_i x_j + \sum_{0<i\leq n} \zeta_i x_i,$$

on a :

$$\begin{aligned}
D_\alpha F &= F(x+\alpha) + F(x) \\
&= \sum_{0<i<j\leq n} \varepsilon_{ij}(x_i+\alpha_i)(x_j+\alpha_j) + \sum_{0<i\leq n} \zeta_i(x_i+\alpha_i) + \sum_{0<i<j\leq n} \varepsilon_{ij}x_ix_j + \sum_{0<i\leq n} \zeta_i x_i \\
&= \sum_{0<i\leq n} x_i \left(\sum_{i<j\leq n} \varepsilon_{ij}\alpha_j \right) + \sum_{0<j\leq n} x_j \left(\sum_{0<i<j} \varepsilon_{ij}\alpha_i \right) + \sum_{0<i<j\leq n} \varepsilon_{ij}\alpha_i\alpha_j + \sum_{0<i\leq n} \zeta_i\alpha_i \\
&= \sum_{0<i\leq n} x_i \left(\sum_{i<j\leq n} \varepsilon_{ij}\alpha_j \right) + \sum_{0<i\leq n} x_j \left(\sum_{0<j<i} \varepsilon_{ij}\alpha_j \right) + \sum_{0<i<j\leq n} \varepsilon_{ij}\alpha_i\alpha_j + \sum_{0<i\leq n} \zeta_i\alpha_i \\
&= \sum_{0<i\leq n} x_i \left(\sum_{i\neq j} \varepsilon_{ij}\alpha_j \right) + \sum_{0<i<j\leq n} \varepsilon_{ij}\alpha_i\alpha_j + \sum_{0<i\leq n} \zeta_i\alpha_i \ .
\end{aligned}$$

Pour avoir une dérivée constante, il est donc nécessaire que $\sum_{0<j\leq n} \varepsilon_{ij}\alpha_j$ soit nulle pour tout i, ce qui donne des contraintes sur α_i. Comme la matrice de représentation de F est donnée par les ε_{ij}, la matrice contenant les contraintes sur les α_i est exactement la matrice représentation.

La seconde inégalité s'obtient par

$$\begin{aligned}
D_\alpha F(x_1,...,x_n) \text{ est constante } &\iff D_\alpha F(x_1,...,x_n) = D_\alpha F(0) \\
&\iff D_\alpha F(x_1,...,x_n) = F(\alpha) + F(0) = F(\alpha) \ .
\end{aligned}$$

□

En combinant ces deux propriétés, on obtient la proposition suivante, valable dans le cas général d'une fonction booléenne de degré 2. Dans tout ce qui suit, les vecteurs seront par défaut des vecteurs lignes.

Proposition 4.3.3. *Une fonction booléenne F de degré 2 est équilibrée si et seulement s'il existe un point α du noyau de la matrice de représentation de F, notée M_f, tel que $F(\alpha)$ soit égal à 1. Formellement :*

$$\begin{aligned}
\textit{F est équilibrée} &\iff \exists \alpha \in \ker(M_f), F(\alpha) = 1 \\
&\iff \exists \alpha \in \ker(M_f), \alpha T_f {}^t\alpha + \lambda \cdot \alpha = 1 \ ,
\end{aligned}$$

où T_f est la matrice triangulaire supérieure telle que $M_f = T_f + {}^tT_f$, et λ est le vecteur correspondant aux termes linéaires de la fonction F.

Démonstration. D'après la proposition 4.3.1,

$$\text{F est équilibrée} \iff \exists \alpha \in \mathbb{F}_2^n, D_\alpha F = 1 \ .$$

Or d'après la proposition 4.3.2,

$$\begin{aligned}
D_\alpha F \text{ est constante } &\iff \alpha \in \ker(M_f) \\
&\iff D_\alpha F = F(\alpha) \ .
\end{aligned}$$

Ainsi,

$$\text{F est équilibrée} \iff \exists \alpha \in \ker(M_f), F(\alpha) = 1 \ .$$

De plus, en notant T_f la matrice triangulaire supérieure telle que $M_f = T_f + {}^tT_f$, et λ le vecteur correspondant aux termes linéaires de la fonction F, on a

$$F(\alpha) = \alpha T_f {}^t\alpha + \lambda \cdot \alpha \ .$$

□

C'est cette formulation qui servira de critère pour déterminer l'équilibre d'une fonction par la suite. Dans le cas particulier de Q_2^3 avec le représentant choisi, cette propriété se simplifie.

Proposition 4.3.4. *Une fonction booléenne F de degré 2 issue d'une combinaison linéaire de parts de Q_2^3 est équilibrée si et seulement s'il existe un point α du noyau de la matrice de représentation de F, notée M_f, tel que $F(\alpha)$ soit égal à 1. Formellement :*

$$\textit{F issue de } Q_2^3 \textit{ est équilibrée} \iff \exists \alpha \in \ker(M_f), \lambda \cdot \alpha = 1 \ .$$

Pour démontrer cette proposition, on va utiliser le lemme suivant :

Lemme 4.1. *Soit M_f la matrice de représentation d'une fonction booléenne de degré 2 à 9 variables, et T_f la matrice telle que $M_f = T_f + {}^tT_f$. Si l'un des blocs non diagonaux de la matrice de représentation est nul, alors $\alpha T_f {}^t\alpha = 0$ pour tout α du noyau de M_f.*

Démonstration. Les matrices de représentation des fonctions booléennes de degré 2 sont de la forme :

0	A	B
A	0	C
B	C	0

et la matrice triangulaire T_f associée est :

0	A	B
0	0	C
0	0	0

Notons $\alpha = (u, v, w)$ un vecteur de $\ker(M_f)$, où u, v et w sont dans $\mathbb{F}_2^3$. On a donc $M_f {}^t\alpha = 0$.

Considérons le cas où le bloc C est nul. Alors l'équation $M_f {}^t\alpha = 0$ impose que $Av + Bw = 0$, $Au = 0$ et $Bu = 0$. On obtient donc que les seules coordonnées non nulles de $T_f {}^t\alpha$ correspondent à $Av + Bw = 0$. Ainsi, $\alpha T_f {}^t\alpha = 0$.

Considérons maintenant le cas où le bloc B est nul. Alors l'équation $M_f {}^t\alpha = 0$ impose que $Av = 0$, $Au + Cw = 0$ et $Cv = 0$. On obtient alors $\alpha T_f {}^t\alpha = {}^tuAv + {}^tvCw$. Or $vC = {}^t({}^tCv) = {}^t(Cv) = 0$, et $Av = 0$. Donc $\alpha T_f {}^t\alpha = 0$.

Considérons enfin le cas où le bloc A est nul. On a alors $Bw = 0, Cw = 0$ et $Bu+Cv = 0$. On obtient donc que $\alpha T_f {}^t\alpha$ est le vecteur de coordonnées $(Bw, Cw, 0)$. Ainsi, $\alpha T_f {}^t\alpha = 0$.

Dans tous les cas, on a donc obtenu que $\alpha T_f {}^t\alpha = 0$. □

La démonstration de la propriété est alors immédiate en utilisant ce lemme.

Démonstration. Dans le cas de Q_2^3 et avec le représentant choisi, les matrices de représentation M_f sont du type :

0	A (bloc ab)	B (bloc ac)
A (bloc ab)	0	0
B (bloc ac)	0	0

D'après le lemme, on obtient bien $\alpha T_f {}^t\alpha = 0$. D'après la propriété 4.3.4, on a donc

$$\text{F issue de } Q_2^3 \text{ est équilibrée} \iff \exists \alpha \in \ker(M_f), \lambda \cdot \alpha = 1 \ .$$

□

Cette dernière propriété de la dérivée permet d'avoir un critère efficace pour déterminer si une fonction issue de Q_2^3 est équilibrée. En effet, appartenir au noyau impose des contraintes sur les α_i. Ces contraintes peuvent être écrites dans une matrice, associée à un système linéaire. Comme démontré dans la preuve ci-dessus, cette matrice est précisément la matrice de représentation de M_f. Pour savoir si une fonction est équilibrée, on cherche à savoir si les contraintes imposées aux éléments du noyau imposent également que la somme des termes linéaires soit nulle. Cela revient à chercher une combinaison linéaire des lignes de la matrice de représentation égale au vecteur représentant les termes linéaires.

Etudions cela sur un exemple, en reprenant la combinaison linéaire non équilibrée $h_1 + g_2 + g_3 + h_3$ présentée en début de section. La matrice de représentation $M_{h_1+g_2+g_3+h_3}$ de la fonction est :

			1	1		1	1	1
			1	1	1	1		
				1		1		1
1	1							
1	1	1						
	1							
1	1	1						
1								
1		1						

et le vecteur correspondant à ses termes linéaires $b_2 + b_3 + c_1 + c_2$ est :

		1	1	1	1

Les points $\alpha = (u_1, u_2, u_3, v_1, ..., w_3)$ appartenant au noyau de $M_{h_1+g_2+g_3+h_3}$ vérifient les équations suivantes, issues de la matrice de représentation vue comme un système linéaire :

$$\begin{cases} v_1 + v_2 + w_1 + w_2 + w_3 = 0 \\ v_1 + v_2 + v_3 + w_1 = 0 \\ v_1 + w_1 + w_3 = 0 \\ u_1 + u_2 = 0 \\ u_1 + u_2 + u_3 = 0 \\ u_2 = 0 \\ u_1 + u_2 + u_3 = 0 \\ u_1 = 0 \\ u_1 + u_3 = 0 \end{cases}$$

Ce système linéaire représente les contraintes imposées aux éléments du noyau de la matrice de représentation.

D'après le lemme 4.2, la dérivée de la fonction par rapport à un point du noyau $\alpha = (u_1, u_2, u_3, v_1, ..., w_3)$ est :

$$D_\alpha F = v_2 + v_3 + w_1 + w_2 \ .$$

On cherche à savoir si les contraintes imposées aux éléments du noyau entrainent que la dérivée, égale à la somme des termes linéaires, est systématiquement nulle. Pour cela il suffit d'observer que la somme des trois premières lignes du système donne exactement la somme des termes linéaires. De façon plus visuelle, cela revient à observer que la somme terme à terme des trois premières lignes de la matrice de représentation est égale au vecteur représentant les termes linéaires. La dérivée est donc toujours nulle, et la fonction $h_1 + g_2 + g_3 + h_3$ n'est pas équilibrée.

La propriété 4.3.4 est donc équivalente à la proposition suivante :

Proposition 4.3.5. *Une fonction booléenne F de degré 2 issue d'une combinaison linéaire de parts de Q_2^3 est équilibrée si et seulement s'il n'existe aucune combinaison linéaire des lignes de la matrice de représentation égale au vecteur représentant les termes linéaires de la fonction.*

Grâce à cette propriété on obtient la caractérisation suivante pour un découpage équilibré de Q_2^3 :

Proposition 4.3.6. *Un découpage correct et indépendant de Q_2^3 est équilibré si et seulement si, pour toute combinaison linéaire F des parts des coordonnées booléennes, le vecteur des termes linéaires de F est linéairement indépendant des lignes de la matrice de représentation de F.*

On peut remarquer que cette propriété est compatible avec le critère d'équilibre sur le représentant de Q_1^3, où la recherche d'équilibre se ramenait à chercher des termes indépendants. La présence de termes indépendants permet d'assurer qu'aucune combinaison linéaire des lignes de la matrice de représentation M_f ne donnera le vecteur des termes linéaires, puisque la colonne correspondant à ce terme indépendant est vide.

4.3.3 Vérifier qu'un découpage de Q_2^3 correct et indépendant est équilibré

Dans cette section, nous proposons un algorithme plus rapide que la recherche exhaustive pour vérifier qu'un découpage correct et indépendant donné est aussi équilibré. En particulier, cet algorithme pourra être utilisé pour vérifier qu'un découpage après ajout de termes correctifs est équilibré. L'idée est ici d'identifier les combinaisons linéaires des parts du représentant de Q_2^3 choisi qui sont susceptibles de ne pas être équilibrées.

Cet algorithme repose sur le constat qu'il existe une forte interdépendance entre certains blocs de la matrice de représentation et les termes linéaires de la forme algébrique normale. Cette interdépendance permet de restreindre les cas à étudier. Rappelons la structure de la matrice de représentation de Q_2^3 :

0	bloc ab	bloc ac
bloc ab	0	0
bloc ac	0	0

Le représentant $(a, ac + b, ab + b + c)$ choisi est à l'origine de l'interdépendance entre les blocs. En effet, quand on identifie les combinaisons linéaires des parts susceptibles de ne pas être équilibrées, on cherche les cas où le vecteur des termes linéaires correspond à une combinaison linéaire des lignes de la matrice. On observe alors que :

- fixer les termes linéaires c_i présents fixe le bloc ab, puisque les termes linéaires c_i proviennent exclusivement de la coordonnée $ab + b + c$;
- comme on cherche les cas où une certaine combinaison linéaire des lignes de la matrice donne le vecteur correspondant aux termes linéaires, les combinaisons linéaires sur le bloc ac doivent permettre d'obtenir les termes linéaires c_i. Fixer les termes linéaires c_i restreint donc les cas possiblement problématiques à étudier en fixant les combinaisons linéaires à l'origine de problèmes ;
- le choix d'une combinaison linéaire sur les lignes du bloc ac fixe une combinaison linéaire sur celles du bloc ab puisque la combinaison linéaire finale doit être définie sur les lignes de la matrice ;
- le choix d'une combinaison linéaire sur le bloc ab fixe une *valeur interdite* sur les termes linéaires b_i, puisqu'on cherche à éviter l'existence d'une combinaison linéaire égale au vecteur des termes linéaires.

Le choix des termes linéaires c_i a donc de fortes répercussions sur les fonctions susceptibles de ne pas être équilibrées. Enumérer les sept cas possibles pour les valeurs de c_i va permettre de passer en revue les cas possiblement problématiques, et de vérifier qu'aucun d'eux ne crée de combinaison linéaire égale au vecteur des termes diagonaux. C'est ce qui est à l'origine de l'algorithme décrit ci-dessous.

Algorithm 1 Verify sharing of Q_2^3

Require: correct and independant sharing.
  for all possible c_i **do**
    fix the corresponding h, i.e. h_j s.t. c_i appears in h_j
    for each linear combination of ac equal to c **do**
      if this linear combination of ab gives b **then**
        return False
      end if
    end for
  end for
  return True

Cet algorithme est plus rapide que la recherche exhaustive car il étudie seulement les cas pouvant devenir problématiques. Comme il y a sept possibilités pour les valeurs de c, et au maximum sept combinaisons linéaires possibles pour trois lignes de la matrice, il n'y a que 49 cas à étudier dans le pire cas, contre 2^9 dans le cas de recherche exhaustive.

4.3.4 Chercher un découpage pour Q_2^3

Afin de trouver un découpage pour Q_2^3, on peut commencer par ajouter des termes correctifs linéaires à une des coordonnées booléennes. Par exemple, on peut corriger $g = ac + b$ avec les termes correctifs b_i. Afin de préserver la propriété de correction, les termes linéaires doivent être ajoutés par paire, tout en préservant l'indépendance. Un terme correctif b_i doit donc être ajouté à la fois à g_{i-1} et à g_{i+1}. Or b_i est aussi le terme indépendant de g_{i+1}, celui qui assure l'équilibre. L'ajout d'un seul terme correctif b_i à deux fonctions entraine donc une perte d'équilibre. Il se produit le même phénomène lorsqu'on essaye de corriger avec deux termes b_i et b_{i+1} en les ajoutant chacun à deux fonctions. En fait, la seule façon de corriger le découpage direct en ajoutant des termes correctifs b_i aux parts g_i, tout en conservant les propriétés de correction et d'indépendance, est d'utiliser trois termes correctifs b_i, b_{i+1}, b_{i-1} en ajoutant $b_{i+1} + b_{i-1}$ aux trois fonctions g_i.

Voici le découpage direct que l'on obtient :

$$
\begin{aligned}
f_1 &= a_2 \\
f_2 &= a_3 \\
f_3 &= a_1 \\
g_1 &= a_2c_2 + a_2c_3 + a_3c_2 + b_2 + b_2 + b_3 = a_2c_2 + a_2c_3 + a_3c_2 + b_3 \\
g_2 &= a_3c_3 + a_3c_1 + a_1c_3 + b_3 + b_1 + b_3 = a_3c_3 + a_3c_1 + a_1c_3 + b_1 \\
g_3 &= a_1c_1 + a_1c_2 + a_2c_1 + b_1 + b_1 + b_2 = a_1c_1 + a_1c_2 + a_2c_1 + b_2 \\
h_1 &= a_2b_2 + a_2b_3 + a_3b_2 + b_2 + c_2 \\
h_2 &= a_3b_3 + a_3b_1 + a_1b_3 + b_3 + c_3 \\
h_3 &= a_1b_1 + a_1b_2 + a_2b_1 + b_1 + c_1 \ .
\end{aligned}
$$

Ce découpage correspond au découpage direct, où les termes b_i ont été placés dans g_{i+1} plutôt que dans g_{i-1}.

Grâce à l'algorithme précédent, on peut vérifier si ce découpage est équilibré.

Remarquons tout d'abord qu'on a une structure cyclique sur les fonctions avec

$$
\begin{aligned}
f_i &= a_{i+1} \\
g_i &= a_{i+1}c_{i+1} + a_{i+1}c_{i-1} + a_{i-1}c_{i+1} + b_{i-1} \\
h_i &= a_{i+1}b_{i+1} + a_{i+1}b_{i-1} + a_{i-1}b_{i+1} + b_{i+1} + c_{i+1} \ .
\end{aligned}
$$

Cela va permettre de simplifier l'étude de l'équilibre en ne considérant que le poids des mots de $\mathbb{F}_2^3$, puisque tout mot de $\mathbb{F}_2^3$ de poids w donné peut-être obtenu par rotation du mot ayant les w premières coordonnées égales à 1 et les suivantes nulles.

Dans ce qui suit, on notera β et γ les vecteurs associés aux coefficients des termes linéaires b_i et c_i respectivement. De plus, rappelons que le bloc ac, où les termes correspondant

à la fonction g_1 sont en rouge, les termes correspondants à g_2 en bleu, et les termes correspondant à g_3 en vert, est de la forme :

1	1	1
1	1	1
1	1	1

Considérons d'abord le vecteur γ de poids 3, i.e. $\gamma = (1, 1, 1)$. Le bloc ab correspondant à la fonction h fixée par ce choix de γ est :

1	1	1
1	1	1
1	1	1

Les combinaisons linéaires possibles pour obtenir γ à partir des lignes de la matrice associée à une combinaison linéaire des g_i sont celles données par $g_1 + g_2 + g_3$ en choisissant une seule ligne, ou bien la somme des trois lignes. Dans ces deux cas, la valeur interdite obtenue pour β correspondant aux termes linéaires b_i est aussi $(1, 1, 1)$. Or pour les valeurs de g et de h choisies, le b obtenu sera systématiquement le vecteur nul, puisque $g_1+g_2+g_3+h_1+h_2+h_3 = g+h = ab+ac+c$. Autrement dit, toutes les combinaisons des parts ayant des termes linéaires $c_1 + c_2 + c_3$ sont équilibrées.

Considérons maintenant le vecteur γ de poids 2, où le 0 est en position j. Les blocs de h fixés par ce choix sont de la forme :

1	1	0
1	1	1
0	1	0

0	0	1
0	1	1
1	1	1

1	1	1
1	0	0
1	0	1

TABLE 6 – Blocs correspondant à $\gamma = (1, 1, 0)$ (à gauche), $\gamma = (0, 1, 1)$ (au milieu) et $\gamma = (1, 0, 1)$ (à droite).

Les combinaisons linéaires des parts de h permettant d'obtenir cette valeur sont les $h_j + h_{j+1}$. Les différentes fonctions et combinaisons linéaires possibles sont les suivantes :

- La fonction g_{j-1} avec la ligne $j + 1$. Cette ligne étant de poids 2, le β interdit sera de poids 2, alors que le β effectivement obtenu par $g_{j-1} + h_{j+1} + h_j$ sera de poids 1 ou 3 du fait de la présence de trois termes linéaires b_k ;
- La fonction g_{j-1} avec les lignes j et $j + 1$. Le β interdit sera alors de poids 1, avec le 1 en position $j + 1$. Or le β obtenu est le même. On obtient donc une combinaison linéaire des lignes de la matrice donnant le vecteur de la diagonale, et donc un cas non équilibré.

Bien que prometteur, ce découpage n'est donc pas compatible avec les implémentations à seuil. Cependant l'étude de ce découpage donne une idée sur les corrections envisageables, et un exemple de la façon dont l'algorithme de vérification fonctionne.

4.4 La classe affine Q_3^3

Pour le cas Q_3^3, les auteurs de [BNN+15] ont montré par recherche exhaustive qu'aucun découpage n'est équilibré. Cependant une telle recherche exhaustive n'est applicable qu'au cas $n = 3$, mais pas au-delà du fait de sa complexité. Le but de cette section est de trouver un algorithme plus rapide pour parvenir à ce résultat avec $n = 3$, avec pour objectif de l'appliquer par la suite à des cas plus grands.

Rappelons que le représentant utilisé pour la classe Q_3^3 est

$$(f, g, h) = (ac + b, ab + b + c, bc + a + b + c) \ .$$

Ce représentant met en jeu trois types de termes quadratiques différents, le critère lié à l'indépendance sera donc inutilisable comme pour Q_2^3. En revanche, les notions de matrice de représentation et de dérivée restent utilisables, en particulier la propriété générale sur l'équilibre de fonctions booléennes de degré deux est conservée. Rappelons cette propriété :

Proposition 4.4.1. *Une fonction booléenne F de degré 2 est équilibrée si et seulement s'il existe un point α du noyau de la matrice de représentation de F, notée M_f, tel que $F(\alpha)$ soit égal à un. Formellement :*

$$\begin{aligned} \textit{F est équilibrée} &\iff \exists \alpha \in \ker(M_f), F(\alpha) = 1 \\ &\iff \exists \alpha \in \ker(M_f), \alpha T_f {}^t\alpha + \lambda \cdot \alpha = 1 \ , \end{aligned}$$

où T_f est la matrice triangulaire supérieure telle que $M_f = T_f + {}^tT_f$, et λ est le vecteur correspondant aux termes linéaires de la fonction F.

Cette propriété peut être utilisée pour prouver plus rapidement que par recherche exhaustive que le représentant de Q_3^3 ne peut pas être corrigé avec des termes linéaires. Pour cela, il faut pouvoir modéliser l'ensemble des termes linéaires possibles.

4.4.1 Modélisation de l'ensemble des termes linéaires possibles

On cherche à représenter l'ensemble des termes linéaires possibles pour les parts d'une coordonnée booléenne. D'après la propriété d'indépendance, une fonction d'indice i sera indépendante des termes d'indice i, ce qui impose une première restriction sur les termes linéaires possibles. Par ailleurs, la propriété de correction impose que la somme des trois parts soit égale à la composante booléenne, ce qui implique que les termes apparaissant, ou n'apparaissant pas dans la forme algébrique normale soient présents un nombre impair, ou pair de fois sur les trois fonctions respectivement. Enfin, pour assurer l'équilibre de chaque part, le terme indépendant de la composante booléenne doit être présent dans chaque part. Ainsi, chaque part f_i doit avoir un terme linéaire qui contient une variable b_j, sinon elle n'est pas équilibrée. Le fait que le nombre de parts de f soit impair impose alors que chaque f_i ne contienne qu'une seule variable de type b_j. Autrement dit, les variables b_j sont réparties soit en suivant la convention du découpage direct, soit en suivant la convention inverse (voir section 4.3.4 sur les découpages de Q_2^3).

L'ensemble des termes linéaires possibles des parts du représentant de Q_3^3 choisi sont donc :

$$
\begin{aligned}
f_{l1} &= \alpha_2 a_2 + \alpha_3 a_3 + \beta b_2 + \overline{\beta} b_3 + \gamma_2 c_2 + \gamma_3 c_3 \\
f_{l2} &= \alpha_1 a_1 + \alpha_3 a_3 + \overline{\beta} b_1 + \beta b_3 + \gamma_1 c_1 + \gamma_3 c_3 \\
f_{l3} &= \alpha_1 a_1 + \alpha_2 a_2 + \beta b_1 + \overline{\beta} b_2 + \gamma_1 c_1 + \gamma_2 c_2
\end{aligned}
$$

$$
\begin{aligned}
g_{l1} &= \alpha_2 a_2 + \alpha_3 a_3 + \beta_2 b_2 + \overline{\beta_3} b_3 + \gamma c_2 + \overline{\gamma} c_3 \\
g_{l2} &= \alpha_1 a_1 + \alpha_3 a_3 + \overline{\beta_1} b_1 + \beta_3 b_3 + \overline{\gamma} c_1 + \gamma c_3 \\
g_{l3} &= \alpha_1 a_1 + \alpha_2 a_2 + \beta_1 b_1 + \overline{\beta_2} b_2 + \gamma c_1 + \overline{\gamma} c_2
\end{aligned}
$$

$$
\begin{aligned}
h_{l1} &= \alpha a_2 + \overline{\alpha} a_3 + \beta_2 b_2 + \overline{\beta_3} b_3 + \gamma_2 c_2 + \overline{\gamma_3} c_3 \\
h_{l2} &= \overline{\alpha} a_1 + \alpha a_3 + \overline{\beta_1} b_1 + \beta_3 b_3 + \overline{\gamma_1} c_1 + \gamma_3 c_3 \\
h_{l3} &= \alpha a_1 + \overline{\alpha} a_2 + \beta_1 b_1 + \overline{\beta_2} b_2 + \gamma_1 c_1 + \overline{\gamma_2} c_2 \ .
\end{aligned}
$$

Le découpage des termes linéaires de chaque coordonnée de Q_3^3 fait donc intervenir sept variables binaires. Cette modélisation des termes linéaires possibles va permettre de représenter facilement les contraintes imposées par le fait que toutes les combinaisons linéaires de parts doivent être équilibrées.

4.4.2 Cas $\dim \ker M_f = 1$

Dans le cas où le noyau de la matrice de représentation est de dimension un, le test d'équilibre implique qu'il n'y a qu'un seul élément α du noyau non nul. En comptant le nombre de termes quadratiques non nuls quand on les évalue en α, on obtient une équation de degré un qui relie les termes linéaires possibles.

Autrement dit, si F est une fonction booléenne de degré 2 avec $F(0) = 0$, et si

$$
F(x_1, ..., x_n) = \sum_{0<i<j\leq n} \varepsilon_{ij} x_i x_j + \sum_{0<i\leq n} \zeta_i x_i \ ,
$$

alors F est équilibrée si et seulement si

$$
\sum_{0<i\leq n} \zeta_i \alpha_i = 1 + \sharp\{(i,j), i<j \text{ tels que } \varepsilon_{ij}\alpha_i\alpha_j \text{ soit non nul}\} \ .
$$

4.4.3 Algorithme pour prouver l'absence de corrections linéaires

Grâce aux notions introduites ci-dessus, on peut prouver plus rapidement que par une recherche exhaustive si, pour un découpage donné des termes quadratiques, (par exemple le découpage direct), il existe une correction n'utilisant que des termes linéaires. Le principe est le suivant : pour chaque combinaison linéaire des f_i, g_i, et h_i telle que la dimension du noyau de la matrice de représentation soit un, on obtient une équation de parité devant être vérifiée par les termes linéaires. En accumulant ces équations, on finit par obtenir une incompatibilité entre ces contraintes, ce qui prouve l'impossibilité de corriger avec des termes linéaires. Dans le cas contraire, on obtient une correction possible. L'algorithme est décrit ci-dessous.

Algorithm 2 Finding linear correction

Require: correct and independent sharing with fixed quadratic terms.
S = linear system
for all linear combination s.t. $\dim \ker M_f = 1$ **do**
compute the linear relation according to the parity
add equation to S
if S has no solution **then**
return False
end if
end for
return solutions from S

4.4.4 Q_3^3 n'a pas de corrections linéaires

En appliquant l'algorithme précedent au représentant de Q_3^3 choisi, on peut obtenir en cinq calculs de noyau une incompatibilité dans les équations vérifiées par les termes linéaires. En effet, il est possible d'exploiter à nouveau la cyclicité des fonctions pour obtenir cinq nouvelles équations, à partir d'une seule équation obtenue par calcul de noyau.

L'implémentation de l'algorithme a été réalisée en Sage. Les cinq combinaisons linéaires dont le noyau a été calculé sont :

$$\begin{aligned} &f_3 + g_2 + h_1 \\ &f_3 + g_2 + h_1 + h_2 \\ &f_3 + g_2 + h_1 + h_3 \\ &f_3 + g_2 + h_1 + h_2 + h_3 \\ &f_3 + g_3 + h_2 + h_3 \ . \end{aligned}$$

Comme énoncé au paragraphe précédent, on peut alors extraire de ces combinaisons linéaires des équations vérifiées par les termes linéaires des parts. On obtient alors cinq équations devant être vérifiées par les termes linéaires des parts. Grâce à la cyclicité, chaque équation permet d'en obtenir cinq autres par décalage circulaire des lignes du système. On obtient donc un système de trente équations linéaires. Ce système n'a pas de solution, ce qui montre que le découpage direct du représentant de Q_3^3 choisi ne peut pas être corrigé avec des termes linéaires.

5 Perspectives

Les travaux envisagés pour la suite du stage sont d'adapter les algorithmes valables pour les boîtes S 3×3 au cas général des classes affines quadratiques des boîtes S $n \times n$.

En particulier, la fonction non-linéaire à 5 variables du nouveau standard SHA-3 (voir [BDPA08] et [SHA15]) est une fonction pour laquelle on ne connaît pas le nombre minimal de parts nécessaires pour obtenir un découpage équilibré. Les coordonnées booléennes de cette boîte S 5×5 sont

$$(ac + b + c, bd + c + d, ce + d + e, ad + a + e, be + a + b) \ .$$

Elle appartient donc bien à une classe affine quadratique, sur laquelle les algorithme développés doivent pouvoir être adaptés. Vue l'importance de cette fonction, il serait intéressant de savoir si sa permutation non-linéaire est compatible avec les implémentations à seuil.

Conclusions

Le stage portait sur l'étude de contre-mesures face aux attaques par canaux auxiliaires liées à l'analyse de la consommation de courant, valides même en présence de glitches. Ces contre-mesures prennent la forme d'un masquage sur les entrées associé à du calcul multi-partite avec le découpage des coordonnées booléennes, restreintes au cas de trois parts. Les résultats par recherche exhaustive de [BNN+15] donnent les classes d'équivalence affine des permutations 3×3 et 4×4, et leur compatibilité ou incompatibilité avec les implémentations à seuil à trois parts. Cependant, la recherche exhaustive de découpage n'est pas faisable pour des boîtes S de taille plus grande. Il est donc nécessaire de trouver des algorithmes permettant de déterminer si les permutations de taille $n \times n$ sont compatibles avec les implémentations à seuil.

Grâce à une meilleure compréhension de la structure mathématiques sous-jacente des différentes classes d'équivalence des boîtes S 3×3, il est possible de développer des algorithmes plus efficaces que la recherche exhaustive pour déterminer si un découpage est valide ou non dans le cas des boîtes S quadratiques. Les critères d'équilibre d'un découpage liés à la présence de termes indépendants, aux liens entre les termes linéaires et la matrice de représentation, et l'étude de fonctions particulières ayant un noyau de dimension zéro ou un sont à l'origine de ces algorithmes. Leur adaptation permet alors d'étudier la compatibilité avec l'implémentation à seuil des boîtes S quadratiques $n \times n$, jusqu'alors inaccessibles.

La fin du stage sera consacrée à l'adaptation de ces algorithmes à des boîtes S de taille plus grande, notamment à celle utilisée par Keccak, afin de tenter de déterminer s'il existe une implémentation à seuil possible n'utilisant que trois parts.

Références

[AG01] Mehdi-Laurent Akkar and Christophe Giraud. An Implementation of DES and AES, Secure against Some Attacks. In *Cryptographic Hardware and Embedded Systems — CHES 2001*, volume 2162 of *Lecture Notes in Computer Science*, pages 309–318. Springer, 2001.

[BDPA08] Guido Bertoni, Joan Daemen, Michaël Peeters, and Gilles Van Assche. Keccak sponge function family main document. Submission to NIST (Round 1), 2008.

[BNN+12] Begül Bilgin, Svetla Nikova, Ventzislav Nikov, Vincent Rijmen, and Georg Stütz. Threshold Implementations of All 3 ×3 and 4 ×4 S-Boxes. In *Cryptographic Hardware and Embedded Systems - CHES 2012*, volume 7428 of *Lecture Notes in Computer Science*, pages 76–91. Springer, 2012.

[BNN+15] Begül Bilgin, Svetla Nikova, Ventzislav Nikov, Vincent Rijmen, Natalia N. Tokareva, and Valeriya Vitkup. Threshold implementations of small S-boxes. *Cryptography and Communications*, 7(1) :3–33, 2015.

[Can16] Anne Canteaut. Lecture Notes on Cryptographic Boolean Functions. www.rocq.inria.fr/secret/Anne.Canteaut/poly.pdf, 2016.

[CJRR99] Suresh Chari, Charanjit Jutla, Josyula Rao, and Pankaj Rohatgi. Towards sound approaches to counteract power-analysis attacks. In *Advances in Cryptology - CRYPTO '99*, volume 1666 of *Lecture Notes in Computer Science*, pages 398–412. Springer, 1999.

[De 07] Christophe De Cannière. *Analysis and Design of Symmetric Encryption Algorithms*. PhD thesis, KU Leuven, 2007.

[GP99] Louis Goubin and Jacques Patarin. DES and differential power analysis (the "duplication" method). In *Cryptographic Hardware and Embedded Systems, CHES'99*, volume 1717 of *Lecture Notes in Computer Science*, pages 158–172. Springer, 1999.

[GU63] René Goscinny and Albert Uderzo. *Astérix et Cléopatre*. Dargaud, 1963.

[Koc96] Paul C. Kocher. Timing Attacks on Implementations of Diffie-Hellman, RSA, DSS, and Other Systems. In *Advances in Cryptology - CRYPTO '96*, volume 1109 of *Lecture Notes in Computer Science*, pages 104–113. Springer, 1996.

[MPO05] Stefan Mangard, Norbert Pramstaller, and Elisabeth Oswald. Successfully attacking masked AES hardware implementations. In *Cryptographic Hardware and Embedded Systems - CHES 2005*, volume 3659 of *Lecture Notes in Computer Science*, pages 157–171. Springer, 2005.

[MS77] Florence MacWilliams and Neil Sloane. *The Theory of Error-Correcting Codes*. North-Holland, 1977.

[NRR06] Svetla Nikova, Christian Rechberger, and Vincent Rijmen. Threshold implementations against side-channel attacks and glitches. In *Information and Communications Security, ICICS 2006*, volume 4307 of *Lecture Notes in Computer Science*, pages 529–545, 2006.

[NRS11] Svetla Nikova, Vincent Rijmen, and Martin Schläffer. Secure hardware implementation of nonlinear functions in the presence of glitches. *Journal of Cryptology*, 24(2) :292–321, 2011.

[SHA15] SHA-3 Standard : Permutation-Based Hash and Extendable-Output Functions. "National Institute of Standards and Technology (NIST), FIPS PUB 202, U.S. Department of Commerce", August 2015.

[Sta10] François-Xavier Standaert. Introduction to side-channel attacks. In *Secure Integrated Circuits and Systems*, pages 27–42. Springer, 2010. perso.uclouvain.be/fstandae/PUBLIS/42.pdf.

A Démontration de la propriété 4.3.1 - Lien entre la dérivée et l'équilibre d'une fonction

Commençons par rappeler la proposition :

Proposition A.0.1. *Une fonction booléenne F de degré 2 est équilibrée si et seulement s'il existe un point α tel que la dérivée de la fonction par rapport à ce point soit égale à 1. Formellement :*

$$F \text{ est équilibrée } \iff \exists \alpha \in \mathbb{F}_2^n, D_\alpha F = 1 \ .$$

Pour démontrer cette propriété, nous aurons besoin d'une définition supplémentaire et d'un lemme associé.

Définition 3. *Soit F une fonction booléenne à n variables. On appelle* biais *de F :*

$$\mathcal{E}(F) = \sum_{x \in \mathbb{F}_2^n} (-1)^{F(x)} = 2^n - 2wt(f) \ .$$

La transformée de Walsh de F est la fonction :

$$a \in \mathbb{F}_2^n \to \sum_{x \in \mathbb{F}_2^n} (-1)^{F(x)+a \cdot x} \ .$$

(les coefficients de Walsh sont les biais des approximations linéaires de F).

Cette définition est reliée à la notion de dérivée comme le montre le lemme suivant :

Lemme A.1. *Pour une fonction booléenne de degré 2 à n variables , on a :*

$$\mathcal{E}(F)^2 = \sum_{\alpha, D_\alpha F = cste} \mathcal{E}(D_\alpha F) \ .$$

Démonstration. On a :

$$\begin{aligned} \sum_{\alpha \in \mathbb{F}_2^n} \mathcal{E}(D_\alpha F) &= \sum_{\alpha \in \mathbb{F}_2^n} \sum_{x \in \mathbb{F}_2^n} (-1)^{F(x+\alpha)+F(x)} \\ &= \left(\sum_{x \in \mathbb{F}_2^n} (-1)^{F(x)} \right) \left(\sum_{y \in \mathbb{F}_2^n} (-1)^{F(y)} \right) \\ &= \mathcal{E}(F)^2 \ . \end{aligned}$$

Dans le cas de fonctions de degré 2, la dérivée est de degré inférieur ou égal à 1. Comme une fonction de degré un est toujours équilibrée, on a dans ce cas $\mathcal{E}(D_\alpha F) = 0$, d'où :

$$\begin{aligned} \sum_{\alpha \in \mathbb{F}_2^n} \mathcal{E}(D_\alpha F) &= \sum_{\alpha, deg(D_\alpha F)=1} \mathcal{E}(D_\alpha F) + \sum_{\alpha, deg(D_\alpha F)=0} \mathcal{E}(D_\alpha F) \\ &= \sum_{\alpha, D_\alpha F = 0} \mathcal{E}(D_\alpha F) + \sum_{\alpha, D_\alpha F = 1} \mathcal{E}(D_\alpha F) \ . \end{aligned}$$

□

La transformée de Walsh et le lemme précedent vont nous permettre de démontrer facilement la propriété.

Démonstration. Montrons d'abord que : F équilibrée $\Rightarrow \exists \alpha, D_\alpha F = 1$.

Si f est équilibrée, alors $\mathcal{E}(F) = 0$. D'après le lemme on a donc :

$$\sum_{\alpha, D_\alpha F = cste} \mathcal{E}(D_\alpha F) = 0$$
$$\sum_{\alpha, D_\alpha F = 0} \mathcal{E}(D_\alpha F) = \sum_{\alpha, D_\alpha F = 1} \mathcal{E}(D_\alpha F) \, .$$

Comme $\mathcal{E}(D_\alpha F) = 2^n$ si la dérivée est identiquement nulle, le terme de gauche est non nul, car $D_0 F = 0$. On obtient qu'il existe au moins un terme α tel que la dérivée par rapport au point α soit 1. On a donc la première implication.

Démontrons maintenant que : $\exists \alpha, D_\alpha F = 1 \Rightarrow F$ équilibrée .

Pour cela, on va d'abord montrer que : $\exists \alpha, D_\alpha F = 1 \Rightarrow \sum_{\alpha, D_\alpha F = cste} \mathcal{E}(D_\alpha F) = 0$.
Grâce au lemme, on obtiendra alors que F est équilibrée.

Comme montré précédemment on a :

$$\sum_{\alpha, D_\alpha F = cste} \mathcal{E}(D_\alpha F) = \sum_{\alpha, D_\alpha F = 0} \mathcal{E}(D_\alpha F) + \sum_{\alpha, D_\alpha F = 1} \mathcal{E}(D_\alpha F)$$

.

Or, d'après la définition de la transformée de Walsh, $\mathcal{E}(D_\alpha F) = 2^n$ si $D_\alpha F = 0$, et $\mathcal{E}(D_\alpha F) = -2^n$ si $D_\alpha F = 1$. On obtient donc :

$$\sum_{\alpha, D_\alpha F = cste} \mathcal{E}(D_\alpha F) = \sum_{\alpha, D_\alpha F = 0} 2^n + \sum_{\alpha, D_\alpha F = 1} -2^n \, .$$

Par ailleurs, l'ensemble $\{\alpha, D_\alpha F = \text{cste}\}$ forme un espace vectoriel, dont $\{\alpha, D_\alpha F = 0\}$ forme un hyperplan, et $\{\alpha, D_\alpha F = 1\}$ un translaté de cet hyperplan. Ces deux ensembles sont donc de même cardinal.

On obtient donc $\sum_{\alpha, D_\alpha F = cste} \mathcal{E}(D_\alpha F) = 0$, puis la proposition grâce au lemme.

□